Чекпоинт 3: Деплой и интерфейс

RAG Чат-бот по роману «Мастер и Маргарита»

Команда bookworm

Горбунов Дмитрий Павлович
Ковалева Дарина Евгеньевна
Тельнов Федор Николаевич
Мацаков Борис Вячеславович

Репозиторий: https://github.com/bookworm-itmo/llm-intro-project

1. Функционал интерфейса

Веб-интерфейс реализован на **Streamlit** и представляет собой чат-бота для ответов на вопросы по роману М.А. Булгакова «Мастер и Маргарита».

Основные возможности

Функция	Описание
Чат-интерфейс	Классический формат диалога с историей сообщений
RAG-поиск	Поиск релевантных фрагментов через FAISS + GigaChat Embeddings
Генерация ответов	OpenRouter (GPT-4o-mini по умолчанию) или Claude Haiku
Источники	Раскрывающийся блок с цитатами из книги под каждым ответом
Реранкер	Переключатель в сайдбаре для BGE Reranker

Элементы интерфейса

1. **Заголовок** — «Чат-бот по роману ‘Мастер и Маргарита’» с пояснением
2. **Сайдбар** — чекбокс «Использовать реранкер» (включен по умолчанию)
3. **История чата** — все предыдущие вопросы и ответы сессии
4. **Поле ввода** — внизу экрана, placeholder «Задайте вопрос о книге»
5. **Спиннер** — индикатор загрузки «Ищу ответ в книге...»
6. **Экспандер источников** — «Источники из книги» с номером главы и текстом

Обработка ошибок

- При отсутствии API-ключей — ошибка при запуске
- При сбое LLM/embedding API — исключение отображается в интерфейсе
- Rate limit — обрабатывается с retry и задержками

2. Скриншот интерфейса

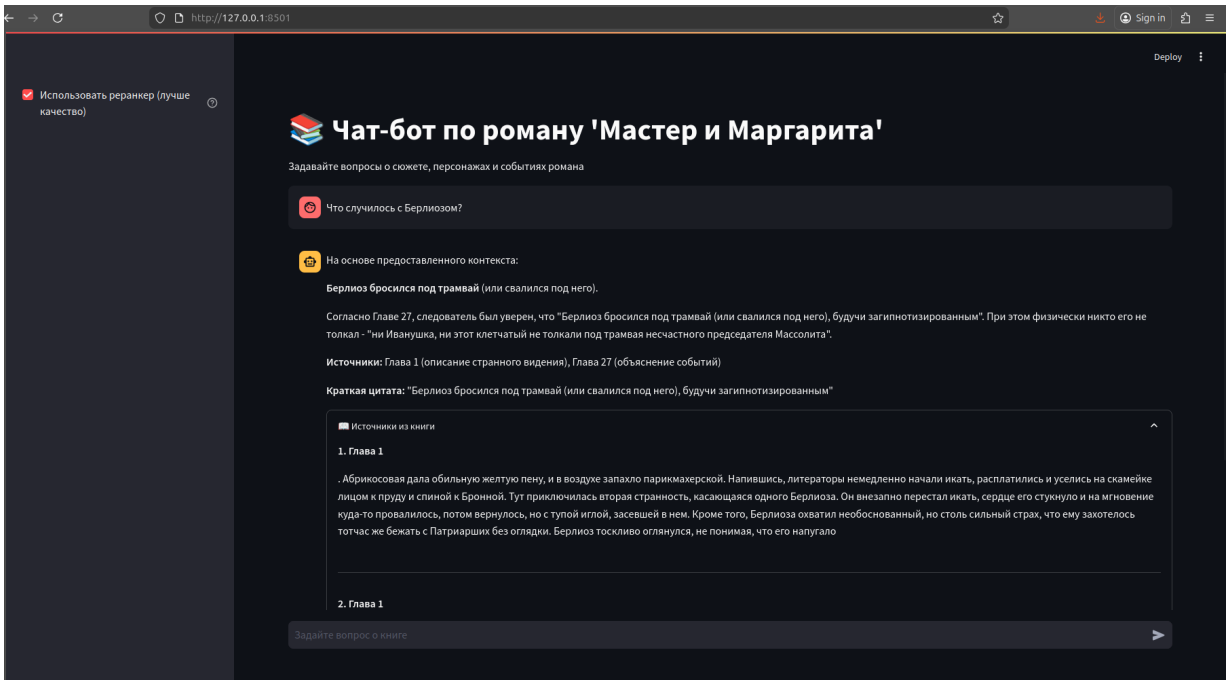

Рис. 1: Интерфейс чат-бота: вопрос «Что случилось с Берлиозом?», ответ с источниками, развернутый блок цитат из глав. Слева — сайдбар с переключателем реранкера.

3. Инструкция по запуску

Требования

- Python 3.10+
- API-ключ OpenRouter (по умолчанию) или Claude (Anthropic)
- API-ключ GigaChat (Сбер) для эмбеддингов

Быстрый старт

```
# Клонировать репозиторий
git clone https://github.com/bookworm-itmo/llm-intro-project
cd llm-intro-project

# Установить зависимости
pip install -r requirements.txt

# Создать .env файл (см. .env.example)
cp .env.example .env
# Заполнить OPENROUTER_API_KEY и GIGACHAT_AUTH_KEY

# Подготовить данные (если нет готовых)
python main.py
```

```
# Запустить интерфейс
streamlit run frontend/app.py
```

Запуск через Docker

```
docker-compose up frontend
```

Структура .env

```
LLM_PROVIDER=openrouter          # или "claude"
OPENROUTER_API_KEY=sk-or-v1-...  # для OpenRouter
OPENROUTER_MODEL=openai/gpt-4o-mini
CLAUDE_API_KEY=sk-ant-...        # для Claude
GIGACHAT_AUTH_KEY=base64_key     # обязательно
```

4. Финальные метрики качества системы

Метрики RAGAS

Метрика	Без реранкера	С реранкером	
Faithfulness	0.856	0.821	-0.035
Answer Relevancy	0.358	0.398	+0.040
Context Precision	0.487	0.474	-0.013
Context Recall	0.431	0.421	-0.010

Таблица 1: Метрики RAGAS на валидационной выборке (70 запросов, GPT-4o-mini)

5. Проведенные эксперименты

Реранкер (BGE-reranker-base)

Реранкер улучшил только answer_relevancy (+0.040), остальные метрики немного просели. Вероятная причина: реранкер переупорядочивает контекст, но для данного датасета это не даёт преимущества — GigaChat эмбеддинги уже хорошо находят релевантные фрагменты.

Что не работало

- **OpenAI Embeddings** — проблемы с VPN
- **GigaChat для LLM** — rate limits (429)

6. Архитектура системы

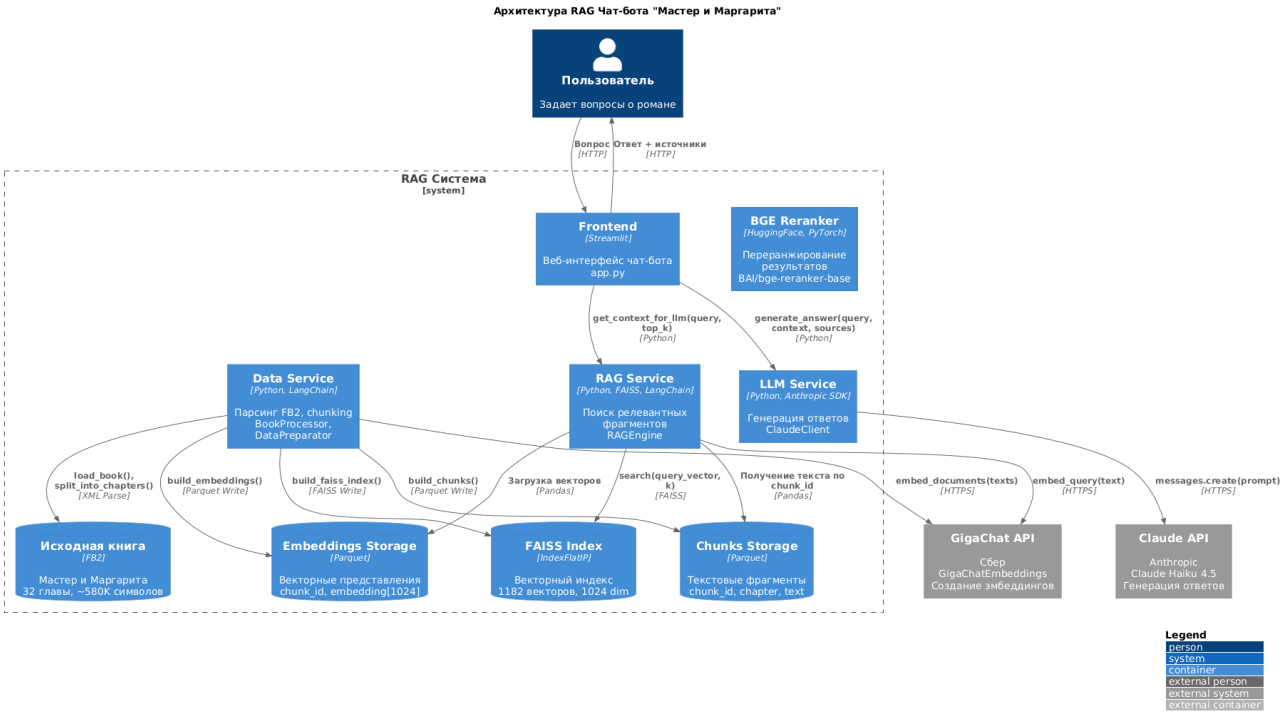

Рис. 2: C4-диаграмма архитектуры RAG чат-бота (PlantUML)

7. Структура репозитория

```
llm-intro-project/
 data/                    # Данные (chunks, embeddings, index)
 services/
    data_service/        # Парсинг FB2, chunking
    rag_service/         # FAISS, embeddings, reranker
    llm_service/         # OpenRouter / Claude API
 frontend/                # Streamlit UI
 validation/              # Скрипты оценки
 metrics/                 # Результаты экспериментов
 docs/                    # Документация
 requirements.txt         # Зависимости (точные версии)
 README.md               # Инструкция по запуску
```

Заключение

Реализован RAG чат-бот с веб-интерфейсом для ответов на вопросы по роману «Мастер и Маргарита». Система использует:

- **GigaChat Embeddings** для векторизации текста (1024 dim)

- **FAISS IndexFlatIP** для поиска
- **BGE Reranker** для переранжирования
- **OpenRouter (GPT-4o-mini)** или Claude для генерации ответов
- **Streamlit** для веб-интерфейса

Система может быть развернута локально через pip или Docker.